**TypeScript Workshop**

# After completing this workshop, student knows how to:

* Configure TypeScript in Visual Studio Code (VSC)
* Set up TypeScript basic types
* Build up a simple web page using TypeScript

<https://www.typescriptlang.org/>

*Create a new folder for these assignments. Place all your code there.*

# Configure and test TypeScript in VSC

Configuring a TypeScript development environment in Visual Studio Code (VSC) involves several steps. TypeScript is a superset of JavaScript that adds static typing to the language, which can be especially helpful in large-scale projects. Here's a step-by-step guide to setting up TypeScript in VSC:

## Step 1: Install Visual Studio Code

If you haven't installed Visual Studio Code yet, you can download it from the official website: [Visual Studio Code](https://code.visualstudio.com/).

## Step 2: Install Node.js and npm

TypeScript requires Node.js and npm (Node Package Manager) for package management. You can download and install Node.js from the official website: [Node.js](https://nodejs.org/).

## Step 3: Install TypeScript

Once Node.js and npm are installed, open a terminal in Visual Studio Code and run the following command to install TypeScript globally (This installs the TypeScript compiler globally on your machine):

npm install -g typescript

## Step 4: Initialize a TypeScript Project

Navigate to the root directory of your project in the terminal and run the following command to initialize a new TypeScript project. This will create a **tsconfig.json** file:

tsc --init

## Step 5: Configure tsconfig.json

Open the **tsconfig.json** file in your project and configure it according to your needs. This file specifies the compiler options for TypeScript. Check, uncomment and correct if needed the following settings:

{

"compilerOptions": {

…

"target": "es5", */\* Set the JavaScript language version for emitted JavaScript and include compatible library declarations. \*/*

"rootDir": "./src", */\* Specify the root folder within your source files. \*/*

"outDir": "./dest", */\* Specify an output folder for all emitted files. \*/*

"removeComments": true, */\* Disable emitting comments. \*/*

"strict": true, */\* Enable all strict type-checking options. \*/*

"noEmitOnError": true, */\* Disable emitting files if any type checking errors are reported. \*/*

},

}

## Step 6: Create a TypeScript file.

Create project source and destination directories (folders) and create a new TypeScript file with a **.ts** extension, for example, **app.ts**, in your *project's source directory*.

## Step 7: Write TypeScript Code

Write some TypeScript code in your **app.ts** file. For example:

function sayHello(name: string): void {

console.log(`Hello, ${name}!`);

}

sayHello("World");

## Step 8: Compile TypeScript to JavaScript

In the terminal, run the following command to compile your TypeScript code to

tsc

This will generate a corresponding **app.js** file in *project destination directory*.

## Step 9: Run your JavaScript code.

You can now run your JavaScript code using Node.js (make sure that you run it in the destination directory:

node app.js

## Step 10: Use Visual Studio Code Extensions

Consider installing Visual Studio Code extensions for TypeScript development. Some popular extensions include:

* **TypeScript and JavaScript Language Features**: This extension provides enhanced TypeScript and JavaScript language support.
* **ESLint**: If you are using ESLint for linting, you can install the ESLint extension.
* **Prettier**: If you are using Prettier for code formatting, install the Prettier extension.

By following these steps, you should have a basic TypeScript development environment set up in Visual Studio Code. Adjust the **tsconfig.json** file and install additional extensions based on your requirements.

Top of Form

# TypeScript Basic types

## Step 1. Declare Variables:

* Declare a variable of type number and assign it a numeric value.
* Declare a variable of type string and assign it a string value.
* Declare a variable of type boolean and assign it either true or false.
* Declare a variable of type any and assign it any value.

## Step 2. Declare Arrays and Tuples:

* Declare an array of numbers and initialize it with at least three numeric values.
* Declare an array of strings and initialize it with at least three string values.
* Declare a tuple representing a person's information with elements for name (string), age (number), and whether the person is a student (boolean).

## Step 3. Declare Functions:

* Declare a function that takes two parameters of type number and returns their sum.
* Declare a function that takes a string parameter and prints it to the console.

## Step 4. Declare Union and Intersection Types:

* Declare a variable that can hold either a number or a string (union type).
* Declare a variable that can hold both a number and a string (intersection type).

## Step 5. Declare Type Aliases:

* Create a type alias for a function that takes two numbers and returns a number (function type).
* Create a type alias for an array that can contain both numbers and strings.

## Step 6. Declare Enums:

* Declare an enum for days of the week.

## Step 7. Declare Type Assertion:

* Use type assertion to convert a variable of type any to a specific type.

## Step 8. Test your code:

* Output above declared variables etc. on the console (console.log).

## 9. Compile TypeScript to JavaScript and run your JavaScript code.

# Building a web page with TypeScript

## Step 1: Project Initialization

1. **Create a Project Folder:**
   * Create a new folder for your project.
2. **Initialize a TypeScript Project:**
   * Open a terminal and navigate to your project folder.
   * Initialize your TypeScript project:

*npm init -y*

*npm install typescript --save-dev*

*tsc --init*

1. **Create TypeScript Configuration:**
   * Create a **tsconfig.json** file in your project folder with the following content:

*{ "compilerOptions": {*

*"target": "es5",*

*"outDir": "./dist",*

*"rootDir": "./src",*

*"strict": true }*

*}*

## Step 2: HTML and CSS

1. **Create HTML File:**
   * Inside the **src** folder, create an **index.html** file:

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>TypeScript Web Page</title>*

*<link rel="stylesheet" href="styles.css">*

*</head>*

*<body>*

*<div id="app"></div>*

*<script src="app.js"></script>*

*</body>*

*</html>*

1. **Create CSS File:**
   * Inside the **src** folder, create a **styles.css** file:

*body {*

*font-family: Arial,*

*sans-serif; margin: 20px;*

*}*

*#app {*

*text-align: center;*

*padding: 20px;*

*background-color: #f2f2f2;*

*border-radius: 10px;*

*}*

## Step 3: TypeScript Code

1. **Create TypeScript File:**
   * Inside the **src** folder, create an **app.ts** file:

## *function greet(name: string): string {*

## *return `Hello, ${name}!`;*

## *}*

## *const appElement = document.getElementById('app');*

## *if (appElement) {*

## *appElement.textContent = greet('TypeScript');*

*}*

## Step 4: Compile TypeScript

1. **Compile TypeScript:**
   * Run the following command to compile your TypeScript code:

*tsc*

1. This will generate a **dist** folder with the compiled JavaScript file (**app.js**).

## Step 5: Open Web Page

1. **Open HTML File:**
   * Open the **index.html** file in a web browser.